Lazy and Eager Relational Learning using Graph-Kernels

Mathias Verbeke[1], Vincent Van Asch[2], Walter Daelemans[2], and Luc De Raedt[1]
[1] Department of Computer Science, KU Leuven, Belgium
{mathias.verbeke,luc.deraedt}@cs.kuleuven.be
[2] Department of Linguistics, Universiteit Antwerpen, Belgium
{vincent.vanasch,walter.daelemans}@uantwerpen.be

Abstract. Machine learning systems can be distinguished along two dimensions. The first is concerned with whether they deal with a feature based (propositional) or a relational representation; the second with the use of eager or lazy learning techniques. The advantage of relational learning is that it can capture structural information. We compare several machine learning techniques along these two dimensions on a binary sentence classification task (hedge cue detection). In particular, we use SVMs for eager learning, and kNN for lazy learning. Furthermore, we employ kLog, a kernel-based statistical relational learning framework as the relational framework. Within this framework we also contribute a novel lazy relational learning system. Our experiments show that relational learners are particularly good at handling long sentences, because of long distance dependencies.

1 Introduction

Solutions to NLP problems require one to take into account both structural information and domain knowledge. Learning systems have essentially two ways of dealing with such information. First, several methods encode the relational information using a set of (automatically or manually) derived features [20]. This effectively propositionalizes the data after which standard machine learning algorithms apply. The drawback of these techniques is that propositionalization results in information loss. On the other hand, (statistical) relational [17] and graph-based learners use the structural information directly, but are often more complex to use. Today there is a growing interest in the use of such statistical relational learning approaches in NLP and several successes have been reported.

Learning techniques can also be distinguished along another dimension that indicates whether they are *eager* or *lazy*. Eager techniques (such as SVMs) compute a concise model from data, while lazy (or memory-based) learners (MBL) simply store the data and use (a variant) of the famous kNN algorithm to classify unseen data. Today, eager methods are much more popular than memory-based ones. Nevertheless, it has been argued [10] that MBL is particularly suited for NLP, since language data contains in addition to regularities, also a lot of subregularities and productive exceptions. Consequently, *lazy* learning may identify these subregularities and exceptions, while *eager* learning often discards them as noise. Thus, lazy learning also has some advantages. It has proven to be successful in a wide range tasks in computational linguistics (e.g., for learning of syntactic and semantic dependencies [21]).

The key contributions of this paper are that we evaluate the performance of learning systems on an NLP task (hedge cue detection) alongst these two dimensions: *propositionalized* versus *relational* representations, and *eager* versus *lazy*

This research is funded by the Research Foundation Flanders (FWO project G.0478.10 - Statistical Relational Learning of Natural Language).

learning. As part of this investigation we also contribute a novel lazy relational learning technique.

A wide range of *eager* learners have been developed and tailored for NLP tasks. Support vector machines (SVM) [7] are one of the most prominent methods, and will be used here as a representative eager learning method. For *lazy* learning we shall employ a kNN framework.

Our relational framework shall be based on kLog [15], a kernel-based statistical relational learning framework, that uses *graphicalization*; a technique that transforms the relational data into a graph-based representation and employs graph-kernels afterwards. The graphicalization does not result in information loss, and is easily understandable. Furthermore, kLog offers a declarative specification of the domain that supports the use of domain knowledge. Our novel lazy relational memory-based learner is based on the kLog representation, and it employs the kLog kernel to define its similarity measure in a memory-based setting.

Thanks to its focus on relations between abstract objects, graph-based relational learning offers the possibility to model a problem on different levels simultaneously, and provides the user with the possibility to represent the problem at the right level of abstraction. For example, sentence classification can be carried out using instances on the token level, without having to resort to a two-step system in which the first step consists of labeling the tokens and the second step is an aggregation step to reach a prediction on the sentence level. Attributes on a higher level, e.g., sentences, can be predicted on the basis of lower level subgraphs, e.g., sequences of tokens, taking into account the relations in the latter, e.g., the dependency tree. This approach has already proven successful for several tasks in NLP [30, 19] and computer vision [1].

Our analysis is performed on a partition of the CoNLL-2010 Shared Task dataset on hedge cue detection, a binary sentence classification task. Sentence classification is particularly interesting for our evaluation as long sentences tend to contain complex dependencies that can be represented with relational representations.

The paper is organized as follows: Section 2 gives an overview of related work. In Section 3 we review kernel-based relational learning with kLog and introduce a new relational memory-based learner. An empirical analysis on the different aspects of the declarative, relational representation is given in Section 4, and discusses the advantages in more depth. Finally, Section 5 concludes.

2 Related Work

Since our analysis comprises a comparison of lazy versus eager learning, both in the relational and propositional setting, the discussion of related approaches is structured along these lines. Since the evaluation uses a dataset from the CoNLL-2010 shared task, state-of-the-art approaches for this problem will be discussed as well.

A wide range of *statistical relational learning* (SRL) systems exist [17]. In principle, many of these are useful to solve problems in computational linguistics. The most popular formalism, Markov Logic, has already been used for tasks such

as coreference resolution [24]. Most SRL systems are based on a combination of learning and inference techniques from probabilistic graphical models.

The technique that we propose is based on kLog[1] [15], a declarative language for kernel-based logical and relational learning with graphs. kLog has two distinguishing features when compared to Markov Logic. First, it employs kernel-based methods grounded in statistical learning theory. Second, it employs Prolog for defining and using background knowledge. As Prolog is a programming language, this is more flexible than the formalism used by Markov Logic. Furthermore, kLogNLP [28] offers a natural language processing module for kLog that enriches kLog with NLP-specific preprocessors, and enables the use of existing libraries and toolkits within this powerful declarative machine learning framework.

A number of approaches have combined relational and instance-based learning. RIBL [12] is a *relational instance-based learning* algorithm that combines memory-based learning with statistical relational learning. It was extended by Horváth et al. [18] to support representations of lists and terms. Armengol and Plaza [2] introduced Laud; a distance measure that can be used to estimate similarity among relational cases, with Shaud [3] as an improvement that is able to take into account the complete structure provided by the feature terms. Ramon [25] proposes a set of methods to perform IBL using a relational representation, and extends distances and prototypes to more complex objects. To the best of the authors' knowledge, these lazy relational learners have not been applied to natural language processing tasks.

In this paper, the classification problem of the *CoNLL-2010 Shared Task*, hedge cue detection, is tackled in an in-domain, closed manner. Hedge cues are words that indicate speculative language. The goal is identifying if a sentence contains such type of words, thus distinguishing factual from uncertain sentences. Since this task involves analyzing the language beyond its propositional meaning, in addition to the lexico-syntactic features of individual words, also the context of the individual words in the sentence or document plays a more important role. This motivates the use of the graph-based relational representation.

During the shared task, Georgescul [16] obtained the best score for the closed task of in-domain Wikipedia hedge cue detection with a macro-averaged F1-score of 75.13%.[2] This score was obtained despite the fact that the system does not use any intricate feature architecture. Each hedge cue of the training set is taken as a feature and prediction occurs directly at the sentence level. For generalization, the set of hedge cues is extended with n-gram subsets of the cues. In a sense, this system resembles a bag-of-words approach. Interestingly, Georgescul [16] also reports the scores for a simple, but effective baseline algorithm: if a test sentence contains any of the hedge cues occurring in the training corpus, the sentence is labeled as UNCERTAIN. This baseline system obtains a macro-averaged F1-score

[1] http://klog.dinfo.unifi.it/

[2] This equals an F1-score on the UNCERTAIN class of 60.17%, but in this paper we prefer reporting the macro-averaged F1-score because it takes the performance on both class labels into account.

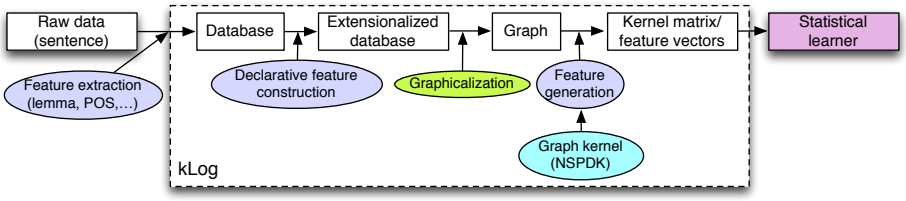

Fig. 1: General kLog workflow.

of 69%. During the shared task and for the Wikipedia data, only the top 3 is able to do better than baseline on the UNCERTAIN class.

3 Kernel-based Relational Learning with Graphs

kLog [15] is a logical and relational language for kernel-based learning, that is embedded in Prolog, and builds upon and links together concepts from database theory, logic programming and learning from *interpretations* (i.e., each interpretation is a set of tuples that are true in the example, and can be seen as a small relational database). It is based on a technique called *graphicalization* that transforms relational representations into graph based ones and derives features from a grounded entity/relationship diagram using graph kernels. This leads to an extended high-dimensional feature space on which a statistical learning algorithm can be applied. The general workflow is illustrated in Figure 1 and will be explained in the following paragraphs.

3.1 Declarative Domain Specification and Feature Construction

Since kLog is rooted in database theory, the modeling of the problem domain is done using an entity-relationship (ER) model [6]. It gives an abstract representation of the interpretations. An example ER model for the hedge cue detection task is given in Figure 3. This ER model is coded declaratively in kLog using an extension of the logic programming language Prolog[3]. Every entity or relationship is declared with the keyword `signature`, which can either be *extensional* or *intensional*. Extensional signatures represent information that is readily available from the input data. An example is the dependency relationship (`depRel`) between two word (`w`) entities, where each relation has its type (`depType`) as a property.

```
signature dependency(word1::w, word2::w, dep_rel::property).
```

On top of these extensional signatures, intensional ones can be defined. In contrast to extensional signatures, intensional signatures introduce novel relations using a mechanism resembling deductive databases. For this type of signatures, due to the declarative nature, no additional preprocessing is required. This type of signatures is mostly used to add domain knowledge about the task at hand. For the hedge cue detection task, the following features provide meaningful additional knowledge [29].

```
signature cw(cw_id::self, lemma::property, pos::property).
cw(CW, L, P) :- w(W,L,P,_,1,_), atomic_concat(c,W,CW).
signature leftof(cw_id::cw, lemma::property, pos::property).
```

[3] The full relational model and data are available at `http://people.cs.kuleuven.be/~mathias.verbeke/klogmbl.html`.

```
leftof(CW,L,P) :- cw(W,_,_), atomic_concat(c,W,CW),
                  next(W1,W), w(W1,L,P,_,_,_).
```

`cw` retains only the words, together with their respective lemma and postag, that appear in a predefined list of hedge cues that was compiled from the training data. `leftOf`, and a similarly defined predicate `rightOf`, also take the two surrounding words of a `cw` in the sentence into account.

3.2 Relational Feature Generation

Subsequently, these interpretations are *graphicalized*, i.e., transformed into graphs. This can be interpreted as unfolding the ER-diagram over the data. We will now extract features from these graphs using a feature generation technique that is based on Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) [8], a particular type of graph kernel. Informally the idea of this kernel is to decompose a graph into small neighborhood subgraphs of increasing radii $r < r_{max}$. Then, all pairs of such subgraphs whose roots are at a distance not greater than $d < d_{max}$ are considered as individual features. The kernel notion is finally given as the fraction of features in common between two graphs. For the sake of completeness we briefly report the formal definitions.

For a given graph $G = (V, E)$, and an integer $r \geq 0$, let $N_r^v(G)$ denote the subgraph of G rooted in v[4] and induced[5] by the set of vertices $V_r^v \doteq \{x \in V : d(x, v) \leq r\}$, where $d(x, v)$ is the shortest-path distance between x and v. A neighborhood $N_r^v(G)$ is therefore a topological *ball* with center v and radius r.

Formally the relation is defined in terms of neighborhood subgraphs as $R_{r,d} = \{(N_r^v(G), N_r^u(G), G) : d(u, v) = d\}$, that is, a relation $R_{r,d}$ that identifies pairs of neighborhoods of radius r whose roots are exactly at distance d. Finally:

$$\kappa_{r,d}(G, G') = \sum_{\substack{A,B \in R_{r,d}^{-1}(G) \\ A',B' \in R_{r,d}^{-1}(G')}} \mathbf{1}_{A \cong A'} \cdot \mathbf{1}_{B \cong B'} \tag{1}$$

where $R_{r,d}^{-1}(G)$ indicates the multiset of all pairs of neighborhoods of radius r with roots at distance d that exist in G, and where $\mathbf{1}$ denotes the indicator function and $\cong$ the isomorphism between graphs.

The NSPDK graph kernel is illustrated in Figure 2 for a distance of 4 between two roots of the neighborhood subgraphs and varying radii. In this toy example, the graph kernel takes a graphicalized sentence parse tree as input, and outputs the subgraphs on the right as (a subset of the) resulting features. This yields a high-dimensional feature space that is much richer than most of the other direct propositionalization approaches, as the relations are explicitly encoded. The result is a propositional learning setting, which enables the use of this set of features in any statistical learner.

3.3 Relational Memory-based Learning

In order to construct a relational memory-based learner, the relational information constructed with kLog and MBL are combined, using the NSPDK

[4] A graph is *rooted* when we distinguish one of its vertices as *root*.

[5] In a graph G, the *induced-subgraph* on a set of vertices $W = \{w_1, \ldots, w_k\}$ is a graph that has W as vertex set and contains every edge of G whose endpoints are in W.

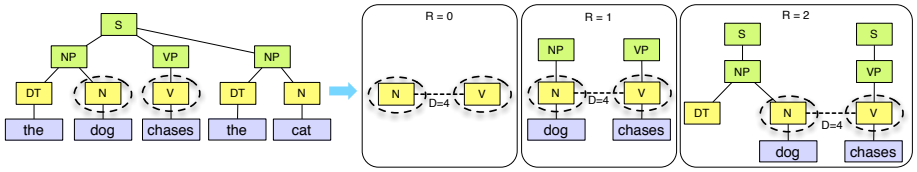

Fig. 2: Illustration of the NSPDK feature concept. Left: two root nodes at distance 4 highlighted; Right: (a subset of) the resulting features for radius 0; radius 1; radius 2.

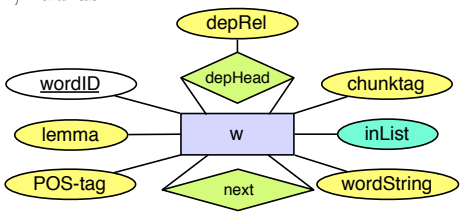

Fig. 3: ER-diagram modeling the hedge cue detection task. Attributes are represented as oval nodes.

Table 1: Number of sentences per class in the training, downsampled training and test partitions (CoNLL-ST 2010, Wikipedia dataset).

Wikipedia	Train	TrainDS	Test
CERTAIN	8,627	2,484	7,400
UNCERTAIN	2,484	2,484	2,234
total	11,111	4,968	9,634

graph kernel as a relational distance measure. The similarities between the instances are readily available from the kernel matrix (also known as the *Gram matrix*), which is calculated by the graph kernel, and thus can be exploited efficiently. A kernel K can be easily transformed into a distance *metric*, using $d_K(x,y) = \sqrt{K(x,x) - 2K(x,y) + K(y,y)}$. This will be referred to as *kLog-MBL*. We both employed a regular kNN setup [14], referred to as *kLog-MBL (NW)*, as well as a distance-weighted variant [11], referred to as *kLog-MBL (W)*. In the latter, a neighbor that is close to an unclassified observation is weighted more heavily than the evidence of another neighbor, which is at a greater distance from the unclassified observation.

4 Advantages of the Relational Representation

In order to illustrate the different steps and the respective advantages of the relational representation, we will evaluate our approach on the CoNLL 2010 Shared Task dataset (Section 4.1). In Section 4.2, we will describe the details of the baseline and benchmarks that were used in the comparison, before turning to an in-depth discussion of the characteristics of the relational representation.

4.1 Dataset

The dataset under consideration consists of sentences from Wikipedia articles that were manually annotated [13]. Due to the collaborative nature of Wikipedia, sentences in this dataset show a very diverse structure. A sentence is considered UNCERTAIN if it contains at least one hedge cue, which is referred to as a *weasel* in the context of Wikipedia. The number of instances per class in the training and test partitions are listed in Table 1.

As can be seen from the table, the data is unbalanced. This can lead to different issues for machine learning algorithms [27]. For MBL, the majority class tends to have more examples in the k-neighbor set, due to which a test instance thus tends to be assigned the majority class label. As a result, the majority

class tends to have high classification accuracy, in contrast to a low classification accuracy for the minority class, which affects the total performance and partly obfuscates the influence of the distance measure [26].

Since the goal of this paper is to show the influence of the relational representation and distance measure, we want to reduce the influence of the imbalancedness of the dataset. Several approaches have been proposed to deal with this (i.e., adjusting misclassification costs, learning from the minority class, adjusting the weights of the examples, etc.). One of the two most commonly used techniques to deal with this problem is sampling [5], where the training dataset is resized to compensate for the imbalancedness. We created a downsampled version of the training sets. This was done in terms of the negative examples (the CERTAIN sentences), i.e., we sampled as many negative examples as there are positive examples. We will refer to this dataset as *TrainDS*.

4.2 Baseline and Benchmarks

In this paper, we want to examine the behavior of a system that uses a relational, graph-based representation to classify the sentence as a whole (i.e., using the graphicalization process) and contrast it with lazy and eager learning systems that do not use this extra step. For this reason, several baselines and benchmarks are included in the result table (Table 2).

The first, simple baseline is a system that labels all sentences with the UNCERTAIN class. This enables us to compare against a baseline where no information about the observations is used.

The first group of benchmarks consists of systems that operate without relational information. These systems typically use a two-step approach; first the individual words in the sentence are classified, whereafter the target label for the sentence is determined based on the number of tokens that are labeled as hedge cues. This requires an extra parameter to threshold this number of individual tokens from which the sentence label is derived (i.e., if more than $X\%$ of the token-level instances are marked as being a hedge cue, the sentence is marked as UNCERTAIN). To optimize this parameter, the training set was split in a reduced training set and a validation set (70/30% split). The influence of this parameter is discussed in more detail in Section 4.5.

The Tilburg Memory-based Learner[6] (TiMBL), a software package implementing several memory-based learning algorithms, among which IB1-IG, an implementation of k-nearest neighbor classification with feature weighting suitable for symbolic feature spaces, and IGTree, a decision-tree approximation of IB1-IG. We will use it in the same setup and with the same feature set as Morante et al. [22]. They used a 5% percentage threshold for sentences, however, our optimization procedure yielded better results with a 30% threshold. In the result table, these variants are referred to as *TiMBL (5%)* and *TiMBL (30%)*.

In order to parameterize TiMBL for the word classification, we used paramsearch[7] [4], a wrapped progressive sampling approach for algorithmic parameter

[6] `http://ilk.uvt.nl/timbl/`

[7] `http://ilk.uvt.nl/paramsearch/`

Table 2: General evaluation of the different systems on the CoNLL-2010 ST Wikipedia corpus with downsampled training set. All scores are macro-averaged.

	Baseline	TiMBL (5%)	TiMBL (30%)	SVM	kLog-MBL(NW)	kLog-MBL(W)	kLog-SVM
Precision	11.59	65.65	69.45	73.03	73.02	73.02	75.37
Recall	50.00	67.83	76.76	79.00	75.24	75.24	73.99
F1-score	18.82	50.92	69.34	74.59	73.96	73.96	74.63

optimization for TiMBL. The IB1 algorithm was chosen as optimal setting, which is the standard MBL algorithm in TiMBL.

The same is done with SVMs[8]. In a first step the (non-graphicalized representation of the) data is converted into binary feature vectors[9]. Subsequently, the SVMs are optimized in terms of the cost parameter C using a grid search with 10-fold cross-validation on the reduced training set. Hereafter the percentage threshold was optimized on the validation set. The SVM without relational information is referred to as *SVM* in the result tables.

We contrasted these systems with a lazy and eager learning approach that use the graph-based relational representation from kLog. For *kLog-SVM*, we used an SVM as statistical learner at the end of the kLog workflow. The results were obtained using the model and graph kernel hyperparameter settings from our previous work [29], for which the cost parameter of the SVM was optimized using cross-validation on the downsampled training set.

The second pair of relational systems use memory-based learning, as discussed in Section 3.3. The value of k was optimized using the reduced training and validation set as discussed above.

4.3 Performance

Table 2 contains the macro-averaged F1-scores of these seven systems. Looking at the tables, one may conclude that all systems perform better than the UNCERTAIN baseline.

The systems are best compared in a pairwise manner. A first interesting observation is that the memory-based learners that use a relational representation (i.e., *kLog-MBL*), perform significantly better than those that use a relational approach, i.e., the *TiMBL* systems. The weighted and unweighted variants of *kLog-MBL* score equally well.

For the SVM setups, *SVM* and *kLog-SVM* score equally well. At first sight, the relational representation does not seem to add much to the performance of the learner. However, when comparing the relational approach on the full (unbalanced) dataset to an *SVM* using the propositional representation, *kLog-SVM* performs significantly better (Table 3). Furthermore, when comparing the results of the regular *SVM* on the balanced and full dataset, a decrease in performance is observed, which is not present for the *kLog-SVM* setup. This indicates that the relational representation increases the generalization power of the learner. We also compared the *kLog-SVM* setup to the best scores obtained during the CoNLL 2010 Shared Task, *viz.* Georgescul [16], in which *kLog-SVM* is able to outperform the state-of-the-art system. This can be attributed to the relational

[8] We used the implementation from scikit-learn [23]

[9] The exact implementation is available at `http://www.cnts.ua.ac.be/~vincent/scripts/binarize.py`

Table 3: Comparison of kLog-SVM with the state-of-the-art results and an SVM using the propositional representation on the full dataset. All scores are macro-averaged.

	Georgescul	SVM	kLog-SVM
Precision	79.29	81.59	77.27
Recall	72.80	68.96	74.16
F1-score	75.13	72.17	75.48

Table 4: Relational feature ranking.

#	Feature	Score	Triplet	Score
1	CW	27.20	cw-next-cw	49.79
2	RightOf	6.69	cw-dh-cw	49.73
3	LeftOf	4.30	cw-dh-word	37.29
4	Next	4.02	cw-next-word	31.79
5	DH	3.42	word-dh-word	12.62
6	WString	-2.35	word-next-word	-5.76
6	InList	-2.35		
6	Chunk	-2.35		
6	Lemma	-2.35		
6	PoS	-2.35		

representation, which offers the possibility to model the sentence as a whole and perform the classification in a single step (i.e., avoiding the need for a two step approach where first token-based classification is performed followed by a thresholding step to obtain the sentence-level classification). We will study this effect in more detail in Section 4.5. In addition, the relational representation is able to model the relations between the words in the sentence explicitly. The graph kernel thus seems to provide a good way to translate the context of the words in a sentence.

4.4 Relational Regularization and Feature Ranking

The importance of the relational features can now be estimated using kLog's relational regularization and feature ranking methods [9], which lift regularization and feature selection to a relational level. The techniques use the relational structure and topology of the domain. Based on a notion of locality, relevant features in the ER-model are tied together. It enables to get deeper insights into the relative importance of the elements in the ER model of the domain. As the added declarative features (`CW`, `LeftOf` and `RightOf`) showed a clear improvement in the results, it is to be expected that these relational features are the main discriminative predicates, while the propositional lexico-syntactic features should be less informative. When measured for kLog-SVM on the full dataset, the results in Table 4 are obtained. In addition, the method also enables to rank the importance of predicate triplets. The right hand side of Table 4 lists the ranking of the possible triplets of predicates of type entity - relationship - entity. Pairs of consecutive hedged words and hedged words that are linked by a dependency relation are clearly very informative relational features.

4.5 Level of Abstraction

The graph-based representation has the advantage that attributes on a higher level, e.g., sentences, can be predicted on the basis of lower level subgraphs, e.g., tokens. It furthermore enables taking into account the relations in the latter, e.g., the dependency tree. This leads to a one-step classification, without the need for an additional thresholding parameter to go from the lower-level classification (e.g., the classification of the individual tokens) to the higher level (e.g., the sentences). The goal of this section is to show when sentence-based systems are more fit for the task than token-based systems.

The baseline system predicts only one type of class label, namely the minority class. The other systems label sentences with both labels and apart from the

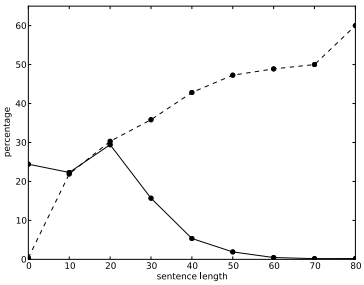

Fig. 4: Fraction of sentences in the Wikipedia test corpus with a given sentence length (—) and the proportion of UNCERTAIN sentences in each bin (- -).

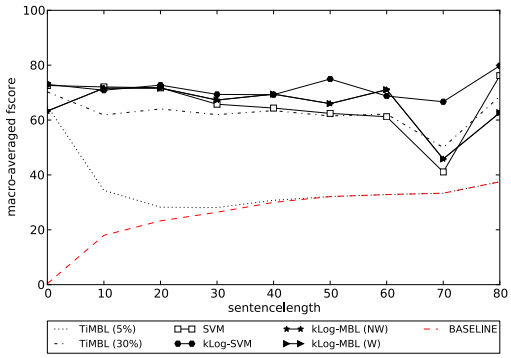

Fig. 5: Macro-averaged F1-score as a function of sentence length, expressed in number of tokens.

observation that one system is more inclined to assign the CERTAIN label than the other, the general scores are not of much help to get more fundamental insights. For this reason, an extra dimension is introduced, namely sentence length. It is an intuitive dimension and other dimensions, like the number of uncertainty cues, are indirectly linked to the sentence length. Figure 5 shows the *evolution* of the macro-averaged F1-score when the sentences to be labeled contain more tokens. To create this figure, the sentences are distributed over 9 bins centered on the multiples of 10. The last bin contains all larger sentences.

Figure 4 shows the fraction of the corpus that is included in each bin (solid line) and the fraction of sentences in each bin that is labeled as UNCERTAIN (dashed line). There are fewer long sentences and long sentences tend to be labeled as UNCERTAIN. As a sanity check, we can look at the behavior of the baseline system in Figure 5. The observation of an increasing number of UNCERTAIN sentences with increasing sentence length (Figure 4) is consistent with the increasing F1-score for the baseline system in Figure 5.

A more interesting observation is the curve of *TiMBL (5%)*, that quickly joins the baseline curve in Figure 5. Although this system performs significantly better than the baseline system, it behaves like baseline systems for longer sentences. Because a large fraction of the sentences is short, this undesirable behavior is not readily noticeable when examining the scores of Table 2. Optimizing the threshold can be a solution to this problem. Changing the threshold influences the chances of a sentence being labeled as UNCERTAIN depending on the sentence length. Increasing the threshold leads to a more unequal distribution of the chances over sentence length. As a result, the behavior of the optimized TiMBL system is more stable with varying sentence length (see *TiMBL (30%)*).

The token-based systems (*SVM* and *TiMBL (30%)*) behave very similar after optimization of the threshold. Indeed, the *SVM* and optimized *TiMBL* curves follow more or less the same course; a course that is different from the other systems. This indicates that by using a two-step approach, the choice of the classifier is of a lesser importance. Although to a more limited extent, this behaviour is also noticeable for *kLog-MBL* in the case of longer sentences. However, when contrasting the kLog-based systems, the curves of the *kLog-SVM*

and *kLog-MBL* systems are mutually divergent for longer sentences, indicating the importance of the classifier.

The importance of the threshold parameter for the propositional, two-step approaches (*TiMBL (5%)*, *TiMBL (30%)* and *SVM*) may be an argument to opt for relational systems. Indeed, the threshold is an extra parameter that has to be learned during training and may introduce errors because of its rigidness. It is the same fixed value for all sentences and it weakens the, possibly positive, influence of the classifier. The kLog-based systems do not require such a threshold and are thus able to *dynamically* look for the best prediction on sentence level using the dependencies between the separate tokens.

The claim that dynamically looking for the best prediction on sentence level is better, is based on the observation that, in general, the *kLog*-based systems perform better than their non-relational counterparts. For the dataset under consideration, the *SVM* system performs not significantly different in F1 than the *kLog-SVM* system, but if we look at their behavior in Figure 5 we see that for almost all sentence lengths *kLog-SVM* performs better. Furthermore, as shown in Section 4.3, *kLog-SVM* generalizes better to the unbalanced version of the dataset when compared to *SVM*, and also obtains the most stable predictions across all sentence lengths.

5 Conclusions

We have used the task of hedge cue detection to evaluate several types of machine learning systems along two dimensions. The results show that relational representations are useful, especially for dealing with long sentences and capturing complex dependencies amongst constituents. The relational representation also allows for one-step classification, without the need for an additional thresholding parameter to go from word to sentence level predictions. We have shown that the kLog framework can be used in both an eager SVM type of learner and a lazy or memory-based learning framework. Especially useful for natural language is that its declarative representation offers a flexible experimentation approach and more interpretable results. In future work we will investigate a hybrid approach that combines lazy and eager learning in the relational case.

References

1. Antanas, L., Frasconi, P., Costa, F., Tuytelaars, T., Raedt, L.: A relational kernel-based framework for hierarchical image understanding. In: Gimel'farb, G., et al. (eds.) Structural, Syntactic, and Statistical Pattern Recognition, LNCS, vol. 7626, pp. 171–180. Springer Berlin Heidelberg (2012)
2. Armengol, E., Plaza, E.: Similarity assessment for relational CBR. In: Proc. of the 4th Int. Conf. on Case-Based Reasoning. pp. 44–58. Springer, London, UK (2001)
3. Armengol, E., Plaza, E.: Relational case-based reasoning for carcinogenic activity prediction. Artif. Intell. Rev. 20(1-2), 121–141 (2003)
4. van den Bosch, A.: Wrapped progressive sampling search for optimizing learning algorithm parameters. In: Proc. of the Sixteenth Belgian-Dutch Conf. on Artificial Intelligence. pp. 219–226 (2004)
5. Chawla, N.V.: Data mining for imbalanced datasets: An overview. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, pp. 875–886. Springer (2010)
6. Chen, P.P.S.: The entity-relationship model - toward a unified view of data. ACM Trans. Database Syst. 1(1), 9–36 (1976)
7. Cortes, C., Vapnik, V.: Support-vector networks. MLJ 20(3), 273–297 (1995)

8. Costa, F., De Grave, K.: Fast neighborhood subgraph pairwise distance kernel. In: Proc. of the 26th Int. Conf. on Machine Learning,. pp. 255–262. Omnipress (2010)
9. Costa, F., Verbeke, M., De Raedt, L.: Relational regularization and feature ranking. In: Proc. of the 14th SIAM Int. Conf. on Data Mining, (2014)
10. Daelemans, W., van den Bosch, A.: Memory-Based Language Processing. Cambridge University Press, New York, NY, USA, 1st edn. (2009)
11. Dudani, S.A.: The distance-weighted k-nearest-neighbor rule. Systems, Man and Cybernetics, IEEE Transactions on SMC-6(4), 325–327 (1976)
12. Emde, W., Wettschereck, D.: Relational instance-based learning. In: ICML. vol. 96, pp. 122–130 (1996)
13. Farkas, R., Vincze, V., Móra, G., Csirik, J., Szarvas, G.: The CoNLL-2010 shared task: learning to detect hedges and their scope in natural language text. In: Proc. of the Conf. on Computational Natural Language Learning — Shared Task. pp. 1–12. ACL, Stroudsburg, PA, USA (2010)
14. Fix, E., Jr: Discriminatory analysis: Nonparametric discrimination: Consistency properties. Tech. Rep. 4, USAF School of Aviation Medicine, TX, USA (1951)
15. Frasconi, P., Costa, F., De Raedt, L., De Grave, K.: kLog: A Language for Logical and Relational Learning with Kernels. CoRR abs/1205.3981 (2012)
16. Georgescul, M.: A hedgehop over a max-margin framework using hedge cues. In: Proc. of the Conf. on Computational Natural Language Learning. pp. 26–31. ACL, Uppsala, Sweden (2010)
17. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning. MIT (2007)
18. Horváth, T., Wrobel, S., Bohnebeck, U.: Relational instance-based learning with lists and terms. MLJ 43(1-2), 53–80 (2001)
19. Kordjamshidi, P., Frasconi, P., van Otterlo, M., Moens, M.F., De Raedt, L.: Relational learning for spatial relation extraction from natural language. In: Muggleton, S.H., et al. (eds.) Inductive Logic Programming. pp. 204–220. Springer (2012)
20. Kramer, S., Lavrač, N., Flach, P.: Relational data mining. In: Dězeroski, S. (ed.) Relational Data Mining, chap. Propositionalization Approaches to Relational Data Mining, pp. 262–286. Springer-Verlag New York, Inc., New York, NY, USA (2000)
21. Morante, R., Van Asch, V., van den Bosch, A.: Joint memory-based learning of syntactic and semantic dependencies in multiple languages. In: Stevenson, S., Carreras, X., Hajič, J. (eds.) Proc. of the Thirteenth Conf. on Computational Natural Language Learning (CoNLL) - Shared Task. ACL, Boulder, Colorado (2009)
22. Morante, R., Van Asch, V., Daelemans, W.: Memory-based resolution of in-sentence scopes of hedge cues. In: Proc. of the Conf. on Computational Natural Language Learning — Shared Task. pp. 40–47. ACL, Stroudsburg, PA, USA (2010)
23. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
24. Poon, H., Domingos, P.: Joint unsupervised coreference resolution with Markov Logic. In: Proc. of the Conf. on Empirical Methods in Natural Language Processing. pp. 650–659. ACL (2008)
25. Ramon, J.: Conceptuele clustering en instance based leren in eerste orde logica. Ph.D. thesis, Informatics Section, Dept. of Computer Science (2002)
26. Tan, S.: Neighbor-weighted k-nearest neighbor for unbalanced text corpus. Expert Syst. Appl. 28(4), 667–671 (2005)
27. Van Hulse, J., Khoshgoftaar, T.M., Napolitano, A.: Experimental perspectives on learning from imbalanced data. In: Proc. of the 24th Int. Conf. on Machine learning. pp. 935–942. ACM, New York, NY, USA (2007)
28. Verbeke, M., Frasconi, P., De Grave, K., Costa, F., De Raedt, L.: kLogNLP: Graph kernel-based relational learning of natural language. In: Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Annual Meeting of the ACL, Baltimore, Maryland, USA (2014, to appear)
29. Verbeke, M., Frasconi, P., Van Asch, V., Morante, R., Daelemans, W., De Raedt, L.: Kernel-based logical and relational learning with kLog for hedge cue detection. In: Muggleton, S., et al. (eds.) Proc. of the 21st Int. Conf. on Inductive Logic Programming. pp. 347–357. Springer (2012)
30. Verbeke, M., Van Asch, V., Morante, R., Frasconi, P., Daelemans, W., De Raedt, L.: A statistical relational learning approach to identifying evidence based medicine categories. In: EMNLP-CoNLL, Jeju, Korea, 13-14 July 2012 (2012)